Evaluarea performantelor calculatoarelor

Benchmarks

Tematica : Benchmarks

Introducere

- Introducere – reprezentarea informatiei in calculator. Tipuri de calculatoare
- Istoria calculatoarelor. Calculatorul von Neumann.
- Arhitectura calculatoarelor personale/ PC. Istoria familiei de procesoare x86

Din interiorul PC-ului

- Memoria in PC. Memoria cache.
- Echipamente periferice. Control transfer de date. Polling, Intreruperi, DMA.
- Bus-uri si interfete folosite in PC. ISA, PCI, PCIe, IDE/ATA, SCSI, RS232, USB, IEEE 1284, etc

Din exteriorul PC-ului

- Echipamente de stocare date: FD, HDD, SSD, CD/DVD, Flash USB
- Echipamente de intrare-iesire: monitor, tastatura, mouse, interfata grafica, interfete audio, etc
- Identificarea si configurarea resurselor PC

Cresterea performantelor PC-ului

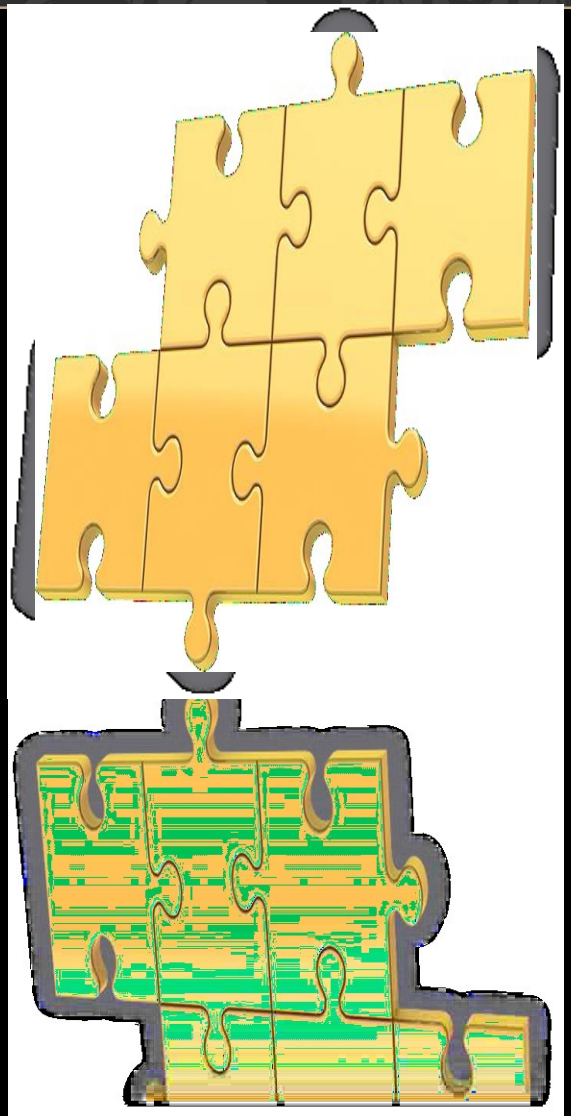
- Clasificarea Flynn. Paralelismul in prelucrarea datelor. CISC vs RISC
- Arhitecturi VLIW, EPIC,Prelucrare secventiala si secvential-paralela
- Pipeline/ Superpipeline. Scalar/Superscalar
- Comparatie arhitecturi procesoare de uz general (GPP), DSP, MicroC, DSC, SoC.

Evaluarea performantelor PC-ului

- **Evaluarea performantelor calculatoarelor. Benchmark-uri.**

11. Evaluarea performantelor calculatoarelor

Obiective



- Prezentarea notiunilor principale despre

modul **cum se pot compara performantele unui PC**

- Familiarizarea cu metodele de evaluare existente

Familiarizarea cu Metricile uzuale folosite la evaluarea performantelor PC-urilor, precum : **Timpul de execuţie , Timpul CPU,**

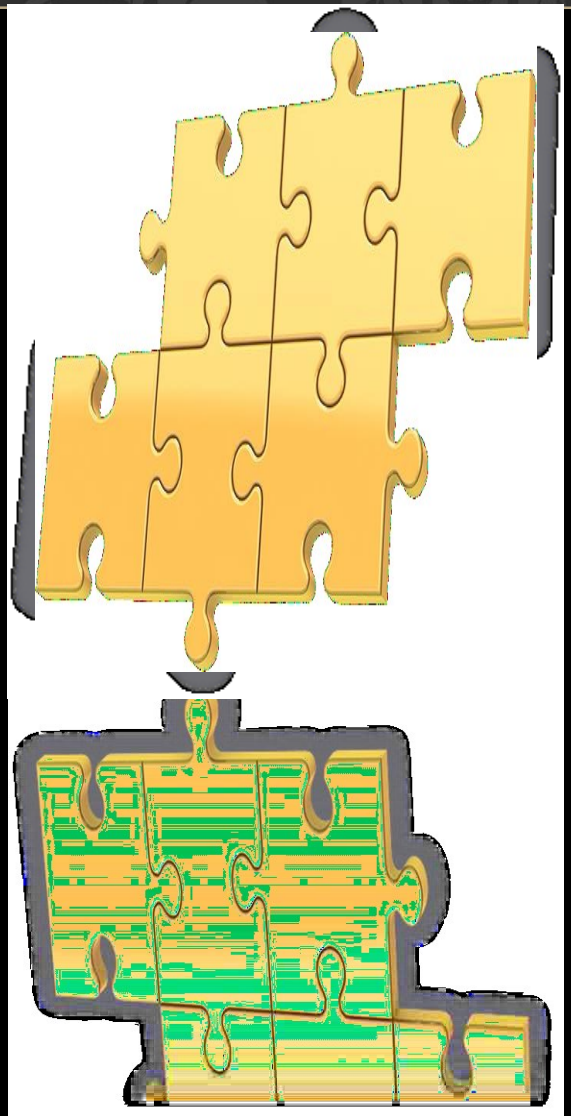
MIPS, MFLOPS, Legea lui Amdahl

si indrumarea spre exemple de calcul folosind acesti indicatori

- urmarirea principalelor aspecte legate de folosirea unor programe reale de evaluare a performantelor PC-urilor

p

Problema de rezolvat



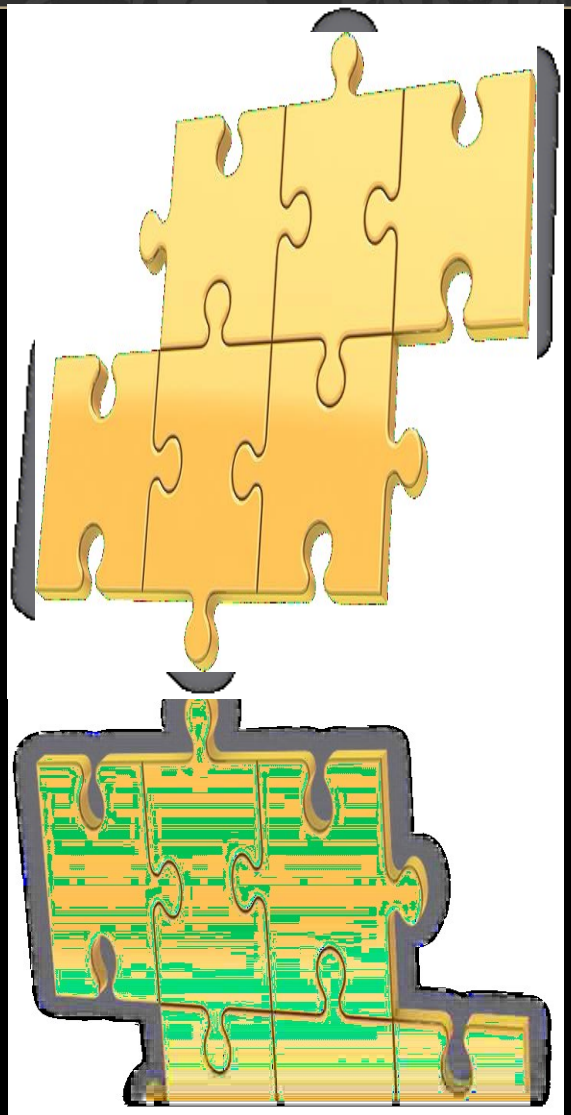
Sistemul A este mai bun decat sistemul B ?

Important dpdv al: inginerului proiectant de sistem, dar si al vanzatorului sau al cumparatorului dintr-o firma, sau al utilizatorului obisnuit

=>cunoasterea si utilizarea unor
Programe reale de evaluare a performantelor
Care se pot baza pe metrici precum:

1. **Timpul de execuţie**
2. **Timpul CPU**
3. **MIPS**
4. **MFLOPS**
5. **Legea lui Amdahl**

Cuprins



Informatia in Evaluarea performantelor calculatoarelor

Metrici folosite

1. Timpul de execuţie

2. Timpul CPU

3. MIPS

4. MFLOPS

5. Legea lui Amdahl

Programe reale de evaluare a performantelor

Evaluarea performantelor calculatoarelor nu poate avea (nu trebuie sa aiba) legatura cu:

Informatia incompleta

"Produsul nostru a fost testat si rezultatele publicate. Compania rivala X nu a publicat rezultatele aceluiasi test, deci produsul nostru e mai bun (adica mai rapid)"

...re - nu exista date concrete !**

Informatia vaga si masuratorile insuficiente sau necorespunzatoare

- folosirea cuvintelor imprecise, precum "mai mult/mare", mai puternic, mai ", aproape, practic (la fel de)" sau alte constructii sinonime, fara asocierea cu date concrete – semnal alarma !

Apelarea la popularitate

"Produsele noastre sunt folosite de 80% din cele mai puternice 100 companii din lume"

- nu inseamna decat ca au un renume asigurat, nu neaparat si ca respectivul produs se potriveste nevoilor utilizatorului care vede reclama

Un sistem cu procesor la 2,5 GHz – inseamna ca e mai bun ca un sistem cu procesor la 2,0 GHz?

NU neaparat

Si atunci ... **Cum se evalueaza performanta calculatoarelor ?** p

Informatia in evaluarea PC-urilor (2)

Performanţa unui calculator nu este uşor de evaluat având în vedere
numărul şi diversitatea arhitecturilor existente pentru sistemele de calcul.

- Mai multe fatete ale performanţei unui calculator:

- din prisma unui utilizator obisnuit:
se masoara performanta pe baza **timpului** necesar pt a executa un anumit program (o sarcina, un task) = $timpul_{executie}$ sau $timpul_{raspuns}$
("Care sistem a terminat mai repede ? A sau B ?")
- din prisma unui inginer (sau al unui manager de callcenter):
performanta prin **banda/debit** (rata=bandwidth, putere de calcul=throughput) : cantitatea totala de sarcini executate in unitatea de timp
("Care sistem a executat mai multe sarcini intr-o zi?")

In [Patterson2013] – analogia cu sistemul de transport aerian:

Care din cele 3 sisteme e mai bun?

- Concorde – viteza cea mai ridicata
- DC-8 – cursa/gama cea mai lunga
- 747 – capacitatea cea mai mare

Performanta – masurata prin diferite aspecte ! Daca se cunosc cele 3 aspecte pentru fiecare din cele 3 sisteme, se poate calcula un coeficient precum ***rata la care avionul transporta pasageri*** = *capacitatea x viteza* (ignorand deci cursa/gama)

- pe baza acestui coeficient se poate oarecum decide care e mai bun

A. Care e cel mai bun sistem dpdv al unui singur pasager ? B. Dar in medie pe un numar mare de pasageri ?

Raspuns: A. Concorde (viteza de cel putin 2x mai ridicata ca 747) , B. 747 deoarece are capacitate mai mare de transport p

11. Evaluarea performantelor calculatoarelor

Informatia in evaluarea PC-urilor (3)

Performanţa unui calculator nu este uşor de evaluat având în vedere
numărul şi diversitatea arhitecturilor existente pentru sistemele de calcul.

- Mai multe fatete ale performanţei unui calculator:

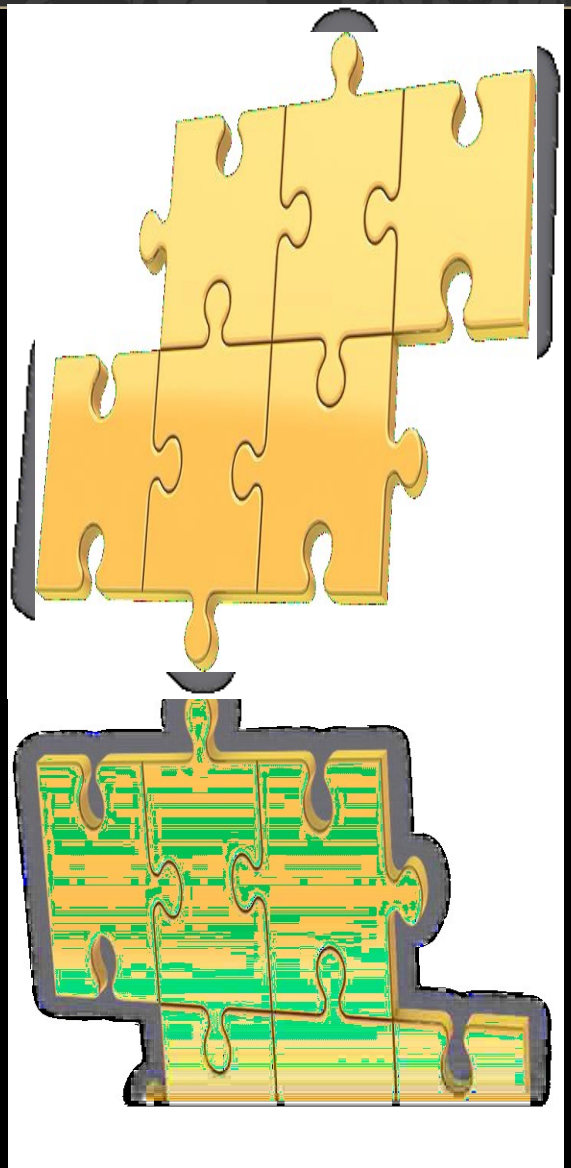
- din prisma unui utilizator obisnuit:
se masoara performanta prin **timpul** necesar pt a executa un
anumit program (o sarcina, un task) = $timpul_{executie}$ sau $timpul_{raspuns}$
("Care sistem a terminat mai repede ? A sau B ?")
- din prisma unui inginer (sau al unui manager de callcenter):
se masoara performanta prin **banda** sau **debit** (bandwidth, putere de calcul=throughput) :
cantitatea totala de sarcini executate in unitatea de timp
("Care sistem a executat mai multe sarcini intr-o zi?")

Dar in general, performanţa este evaluată din prisma timpului
necesar PROCESORULUI să execute o anumita sarcina (un task) predefinita,
- există şi alte metrici (utilizarea memoriei sau consumul de energie)
la fel de importante în anumite aplicaţii.

1. **Timpul de execuţie**
2. **Timpul CPU**
3. **MIPS**
4. **MFLOPS**
5. **Legea lui Amdahl**

11. Evaluarea performantelor calculatoarelor

Cuprins



Informatia in Evaluarea performantelor calculatoarelor

Metrici folosite

1. **Timpul de execuţie**

2. **Timpul CPU**

3. **MIPS**

4. **MFLOPS**

5. **Legea lui Amdahl**

Programe reale de evaluare a performantelor

p

Metrici – timpul de executie

Timpul de execuţie este măsura cel mai des folosită pentru evaluarea performanţelor sistemelor de calcul: calculatorul care obţine timpul de execuţie (măsurat în secunde) cel mai scurt la executarea unui set de operaţii, este considerat cel mai performant.

Timpul de execuţie poate lua diferite forme particulare în măsurarea duratei anumitor evenimente, un astfel de exemplu fiind timpul de răspuns
- definit ca timpul necesar terminării unei sarcini
(de ex. timpul în care sistemul reuşeşte să furnizeze un rezultat) şi cuprinde:
- timpul necesar accesării memoriei,
- timpul necesar operaţiilor de I/O şi
- timpul necesar operaţiilor executate de sistemul de operare (S.O.).

In cazul sistemelor care folosesc multiprogramarea, aceşti timpi se întrepătrund, nemaiputând fi evaluaţi corect => este necesară introducerea unui alt indicator ($timpul_{CPU}$) care să indice timpul efectiv în care CPU execută un program, fără a lua în considerare timpul de aşteptare pentru operaţii de I/O sau timpul în care CPU execută alte programe.
De asemenea, $timpul_{CPU}$ poate fi divizat în:
- $timpul_{CPUutilizator}$ corespunzător execuţiei programului utilizator şi
- $timpul_{CPUsistem}$ necesar execuţiei funcţiilor apelate de S.O.

! ceea ce percepe utilizatorul NU este $timpul_{CPU}$

p

Metrici – timpul de executie (2)

Atunci când se doreşte compararea a două sisteme de calcul (A si B):
- se spune despre sistemul A ca este mai rapid decat sistemul B daca ***timpul de executie al unui program este mai scurt*** pentru sistemul A decat pentru sistemul B:

- "computerul A este de **n** ori **mai rapid** decat computerul B":

$$\frac{t_E(B)}{t_E(A)} = n \quad (1)$$

Astfel, raţionamentul „calculatorul A este **cu n% mai rapid** decât calculatorul B" poate fi exprimat de relaţia:

$$\frac{t_E(B)}{t_E(A)} = 1 + \frac{n}{100} \quad (2)$$

unde t_E = timpul de execuţie.

- timpul de execuţie = inversul performanţei ($t_E = 1/P$) :

$$\frac{t_E(B)}{t_E(A)} = \frac{P(A)}{P(B)} = 1 + \frac{n}{100} \quad (3)$$

Creşterea performanţei (notată *n*) este exprimată ca
diferenţa dintre performanţa calculatorului mai rapid şi cea a calculatorului mai lent, raportată la performanţa calculatorului mai lent:

$$n = \frac{P(A) - P(B)}{P(B)} \cdot 100 = \frac{\frac{E}{t_E(A)} - \frac{E}{t_E(B)}}{\frac{E}{t_E(B)}} \cdot 100 \quad (4)$$

s-a presupus că dintre cele două calculatoare A şi B, cel mai lent este B

Metrici – timpul de executie (3)

[redacted]

- [redacted]

Daca se foloseste **banda**, atunci spunem:

Banda (rata sau puterea de calcul) lui A e de n ori mai mare decat banda lui B

si inseamna ca nr de sarcini rezolvate in unitatea de timp folosind sistemul A este de n ori mai mare decat cele rezolvate folosind sistemul B.

- in gen, n>1 (supraunitar)

- timpul in gen se raporteaza la completarea unui task (sarcini) precum:
  - acces la disc,
  - acces la memorie,
  - activitatea unui periferic I/O

p

Metrici – timpul de executie (4)

Exemplul 00h:

Dacă calculatorul A execută un program în 15 secunde, iar calculatorul B execută acelaşi program în 20 secunde, cu cât va fi mai rapid calculatorul A faţă de calculatorul B?

Soluţie: Se calculează creşterea performanţei după relaţia (4) astfel:

$$n = \frac{P(A) - P(B)}{P(B)} \cdot 100 = \frac{t_E(B) - t_E(A)}{t_E(A)} \cdot 100$$

$$n = \frac{t_E(B) - t_E(A)}{t_E(A)} \cdot 100 = \frac{20 - 15}{15} \cdot 100 = 33 \quad [\%]$$

=> calculatorul A este cu 33 % mai rapid decât calculatorul B.

Exemplul 01h:

Dacă un calculator A rulează un anumit program în 10 secunde și un calculator B rulează același program în 15 secunde, se cere:

a) De câte ori e mai rapid calculatorul A decât calculatorul B?

b) Cu cât e mai rapid calculatorul A decât calculatorul B?

Soluție:

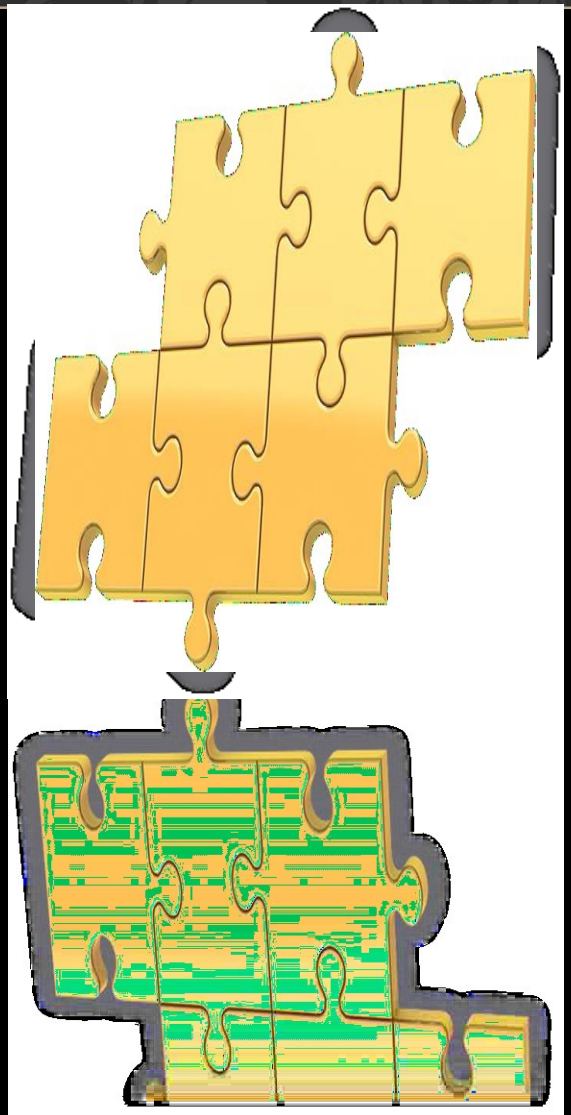
a) $n = \frac{t_{E\,B}}{t_{E\,A}} = \frac{15s}{10s} = 1,5 \Rightarrow n = 1,5$

=> calculatorul A este de 1,5 ori mai rapid decât calculatorul B.

b) $n = \frac{t_{E\,B} - t_{E\,A}}{t_{E\,A}} \cdot 100 = \frac{15-10}{10} \cdot 100 = \frac{5}{10} \cdot 100 = 50 \Rightarrow n = 50\%$

$\Rightarrow$ calculatorul A este cu 50% mai rapid decât calculatorul B.

Cuprins



Informatia in Evaluarea performantelor calculatoarelor

Metrici folosite

1. Timpul de execuţie

2. Timpul CPU

3. MIPS

4. MFLOPS

5. Legea lui Amdahl

Programe reale de evaluare a performantelor

p

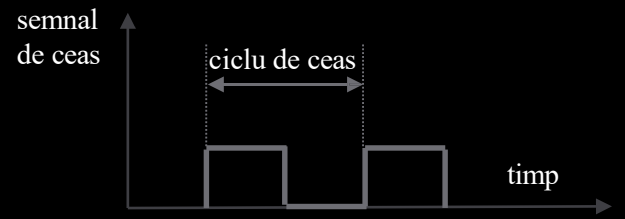
Metrici – timpul de executie al procesorului

Sistemul de calcul foloseşte un semnal de ceas pentru a raporta şi sincroniza execuţia diferitelor operaţii.
- acest semnal de ceas este definit de durata ciclului de ceas (exprimată în secunde) sau de frecvenţa ceasului (exprimată în Hz).

Pentru un program timpul de executie al procesorului ($timpul_{CPU}$) (pt un program) poate fi exprimat:
[secunde/program]

$$t_{CPU} = C_{CPU} \cdot T_C \quad (5)$$

unde C_{CPU} = numărul ciclurilor de ceas ale CPU necesare pt execuţia programului, iar
T_C = durata (perioada) ciclului de ceas [secunde/ciclu ceas].



- pt definirea timpului: ciclul de ceas (clock cycle) = timpul intre 2 fronturi crescatoare (sau descrescatoare) consecutive ale unui semnal de ceas periodic
= masura utilizata pt a specifica timpul necesar executiei unui anumit task

- $timpul_{CPU}$ poate fi exprimat în funcţie de frecvenţa ceasului $f_C = 1/T_C$:

$$t_{CPU} = \frac{C_{CPU}}{f_C} \quad (6)$$

- fie N= numărul de instrucţiuni executate de program
=> CPI = numărul mediu de cicluri de ceas pe instrucţiune:

$$CPI = \frac{C_{CPU}}{N} \quad (7)$$

Din (5) =>

$$t_{CPU} = N \cdot CPI \cdot T_C \quad (8)$$

sau

$$t_{CPU} = \frac{N \cdot CPI}{f_C} \quad (9)$$

Metrici – timpul de executie al procesorului (2)

Uneori este utilă calcularea ***numărului total de cicluri de ceas ale CPU***, acesta obţinându-se din relaţia:

$$C_{CPU} = \sum_{i=1}^{n} (CPI_i \cdot I_i) \quad (10)$$

unde CPI_i = numărul mediu de cicluri de ceas necesar pentru execuţia instrucţiunii i, iar
I_i = numărul de execuţii ale instrucţiunii i într-un program.

=> timpul t_{CPU} din relaţia (5) poate fi exprimat

$$t_{CPU} = T_C \cdot \sum_{i=1}^{n} (CPI_i \cdot I_i) \quad (11)$$

iar nr total de cicluri pe instrucţiune CPI din relatia (6) devine:

$$CPI = \frac{\sum_{i=1}^{n} (CPI_i \cdot I_i)}{N} = \sum_{i=1}^{n} \left(CPI_i \cdot \frac{I_i}{N} \right) = \sum_{i=1}^{n} (CPI_i \cdot F_i) \quad (12)$$

- unde F_i = frecvenţa cu care instrucţiunea i apare în program

Metrici – timpul de executie al procesorului (3)

Exemplul 02h:

Fie frecvenţele de utilizare a instrucţiunilor pe care le poate executa un sistem de calcul cele exprimate în tabelul următor:

Frecvenţele de utilizare a instrucţiunilor

Tip instrucţiune	*Frecvenţa*	*Cicluri ceas*
UAL	50%	1
Încărcare (load)	20%	2
Memorare (store)	10%	2
Salt	10%	3
altele	10%	2

Care va fi numărul total mediu de cicluri pe instrucţiune ?

Soluţie:

Numărul mediu de cicluri pe instrucţiune se calculează cu relaţia (12) astfel:

$$CPI = 1 \cdot 0{,}5 + 2 \cdot 0{,}2 + 2 \cdot 0{,}1 + 3 \cdot 0{,}1 + 2 \cdot 0{,}1 = 1{,}6$$ **[cicluri/instrucţiune]**

deci *în medie*, pentru execuţia unei instrucţiuni, vor fi necesare 1,6 cicluri de ceas.

p

Exemplul 03h

Un program rulează în 10 secunde pe calculatorul A, care are frecvența de ceas de 2GHz. Un alt calculator B, rulează același program în doar 7 secunde. Pentru a obtine acesta reducere a timpului, inginerul proiectant a descoperit că este necesara o creștere a frecvenței ceasului, afectand restul arhitecturii CPU. Daca sistemul B va necesita de 1,4 ori mai multe cicluri ceas decat sistemul A pentru acel program, care este frecvența ceasului pe care ar trebui să o urmărească proiectantul ?

Soluție:

Numărul ciclurilor de ceas necesare pt rularea programului pe calculatorul A este:

$$t_{CPU_A} = \frac{C_{CPU_A}}{f_{C_A}} \Rightarrow 10s = \frac{C_{CPU_A}}{2 \cdot 10^9 \frac{cicluri}{secundă}} \Rightarrow C_{CPU_A} = 10 \cdot 2 \cdot 10^9 cicluri = 20 \cdot 10^9 cicluri$$

Numărul ciclurilor de ceas necesare pt rularea programului pe calculatorul B este de 1,4 ori mai mare.

$$t_{CPU_B} = \frac{1{,}4 \cdot C_{CPU_A}}{f_{C_B}} \Rightarrow 7s = \frac{1{,}4 \cdot 20 \cdot 10^9 cicluri}{f_{C_B}} \Rightarrow f_{C_B} = \frac{1{,}4 \cdot 20 \cdot 10^9 cicluri}{7s} = \frac{0{,}2 \cdot 20 \cdot 10^9 cicluri}{s} \Rightarrow f_{C_B} =$$

$$4 \cdot 10^9 \frac{cicluri}{s}$$

$$\Rightarrow f_{C_B} = 4GHz.$$

Pentru a rula programul în 7 secunde, calculatorul B trebuie să dubleze frecvența de ceas corespunzătoare calculatorului A.

p

Metrici – timpul de executie al procesorului (5)

Exemplul 04h

Fie două implementări ale aceluiași set de instrucțiuni.
Calculatorul A are perioada ciclului de 0,3 ns și executa in medie 2 cicluri ceas pe instructiune pentru un anumit program, iar calculatorul B are perioada ciclului 0,6ns și executa 1,2 cicluri pe instructiune pentru același program.

Care calculator este mai rapid pentru acest program și de câte ori?

Soluție:

Știm că numărul de instrucțiuni al programului este același = N.

$$C_{CPU} = CPI \cdot N \Rightarrow C_{CPU_A} = N \cdot 2$$

$$C_{CPU_B} = N \cdot 1,2$$

$$t_{CPU} = C_{CPU} \cdot T_C = CPI \cdot N \cdot T_C \Rightarrow t_{CPU}(A) = N \cdot 2 \cdot 0,3ns = 0,6N(ns)$$

$$t_{CPU}(B) = N \cdot 1,2 \cdot 0,6ns = 0,72N(ns)$$

$\Rightarrow$ calculatorul A este mai rapid

$$\frac{performanța\ CPU_A}{performanța\ CPU_B} = \frac{t_E(B)}{t_E(A)} = \frac{720N}{600N} = 1,2$$

$\Rightarrow$ calculatorul A este de 1,2 ori mai rapid decât calculatorul B pentru acel program

Exemplul 05h

Un proiectant de compilatoare încearcă să se decidă între două secvențe de cod pentru un anumit calculator. Proiectantul hardware i-a dat următoarele informații:

CPI pt fiecare clasă de instrucțiuni

	A	B	C
CPI	1	2	3

Nr de instrucțiuni pt fiecare clasă de instrucțiuni

Secvența de cod	A	B	C
1	2	1	3
2	4	2	1

Se cere:

a) Care secvență de cod execută mai multe instrucțiuni?

b) Care secvență de cod va fi mai rapidă?

c) Cât este CPI pentru fiecare secvență?

Soluție:

a) Secvența 1 execută: 2 + 1 + 3 = 6 instrucțiuni (=N1)

Secvența 2 execută: 4 + 2 + 1 = 7 instrucțiuni (=N2)

$\Longrightarrow$ secvența 2 execută mai multe instrucțiuni.

b) $C_{CPU} = \sum_{i=1}^{n} (CPI_i \cdot N_i) \Longrightarrow$

$C_{CPU_1} = (2 \cdot 1) + (1 \cdot 2) + (3 \cdot 3) = 2 + 2 + 9 = 13\ cicluri$

$C_{CPU_2} = (4 \cdot 1) + (2 \cdot 2) + (1 \cdot 3) = 4 + 4 + 3 = 11\ cicluri$

$\Longrightarrow$ secvența de cod 2 este mai rapidă, deși execută cu o instrucțiune mai mult.

c)

$$CPI = \frac{C_{CPU}}{N} \Longrightarrow \begin{cases} CPI_1 = \frac{13}{6} = 2,16 \\ CPI_2 = \frac{11}{7} = 1,57 \end{cases}$$

Exemplul 06h

O aplicație (un program) rulează în 20 secunde pe un calculator.
Apare o nouă versiune a aplicației care necesită doar 0,7 instrucțiuni din numărul primei versiuni.
Din păcate, acesta crește valoarea CPI cu 10% din valoarea CPI initiala.

a) Cât de rapid ne aşteptăm să ruleze acestă nouă versiune?
b) Cat timp se castiga cu aceasta noua versiune?

Soluție:

$$t_E = Nnou \cdot CPInou \cdot T_C = 0,7N \cdot 1,1CPI \cdot T_C = 0,7 \cdot 1,1 \cdot 20 \Rightarrow 15,4\ secunde$$

20sec ... 100%
(20-15,4) sec ... X $\Rightarrow$ se castiga 23% din timp

p

Metrici – timpul de executie al procesorului (8)

Exemplul 07h:

Pentru sistemul de calcul din exemplul 02h ->
se pp. că 20% dintre instrucţiunile UAL vor fol un operand încărcat din memorie (este deja încărcat în unul dintre regiştrii şi nu mai e nevoie de o operaţie de Load).

Frecvenţele de utilizare a instrucţiunilor

Tip instrucţiune	*Frecvenţa*	*Cicluri ceas*
UAL	50%	1
Încărcare (load)	20%	2
Memorare (store)	10%	2
Salt	10%	3
altele	10%	2

Ca o metodă de îmbunătăţire a performanţelor acestui sistem de calcul se propune adăugarea unor instrucţiuni UAL care au un operand sursă în memorie (deci nu mai implica si instructiuni de load), dar care vor necesita 2 cicluri de ceas. Se pp. că setul extins de instrucţiuni va creşte cu 1 numărul ciclurilor de ceas pentru execuţia instrucţiunilor de salt, dar nu afectează durata ciclului de ceas.

Care va fi noul număr mediu de cicluri pe instrucţiune CPI_{nou} ? Precizaţi dacă modificarea propusă va creşte performanţa CPU.

Soluţie: sunt cu 20% mai puţine instrucţiuni UAL care nu vor mai avea nevoie nici de încărcarea unor operanzi din memorie (pentru că sunt deja disponibili)
=> se vor elimina şi 20% din instrucţiunile de încărcare (Load).
În schimb, vor fi adăugate 20% instrucţiuni UAL cu regiştrii şi memoria care dureaza 2 cicluri.

Instrucţiunile de memorare contribuie în acelaşi fel ca în exemplul 02h,
însă la instrucţiunile de salt se modifică numărul de cicli din 3 în 4.
=>numărul de instrucţiuni la care se raportează noua valoare se modifică: vor fi cu 20% mai puţine instrucţiuni UAL.

Exemplul 07h: Solutie (continuare)

- se reduc 20% din instruct UAL cu 1 ciclu
  => se reduce si nr instruct Load cu 20%
- instrucţ de memorare si altele contribuie la fel ca in exemplul 02h,
- la instrucţiunile de salt se modifică numărul de cicli din 3 în 4.

+ trebuie adaugate noile instructiuni UAL cu 2 cicli

=> numărul de instrucţiuni la care se raportează noua valoare: cu 20% mai puţine

Frecvenţele de utilizare a instrucţiunilor

Tip instrucţiune	*Frecvenţa*	*Cicluri ceas*
UAL	50%	1
Încărcare (load)	20%	2
Memorare (store)	10%	2
Salt	10%	3
altele	10%	2

$$CPI_{nou} = \frac{1\cdot[0{,}5-(0{,}2\cdot 0{,}5)]+2\cdot[0{,}2-(0{,}2\cdot 0{,}5)]+2\cdot(0{,}2\cdot 0{,}5)+2\cdot 0{,}1+4\cdot 0{,}1+2\cdot 0{,}1}{1-(0{,}2\cdot 0{,}5)}$$

$$CPI_{nou} = \frac{1{,}6}{0{,}9} = 1{,}78 \text{ [cicluri/instrucţiune]}$$

Performanţa sistemului înainte şi după modificarea propusă se va calcula astfel:

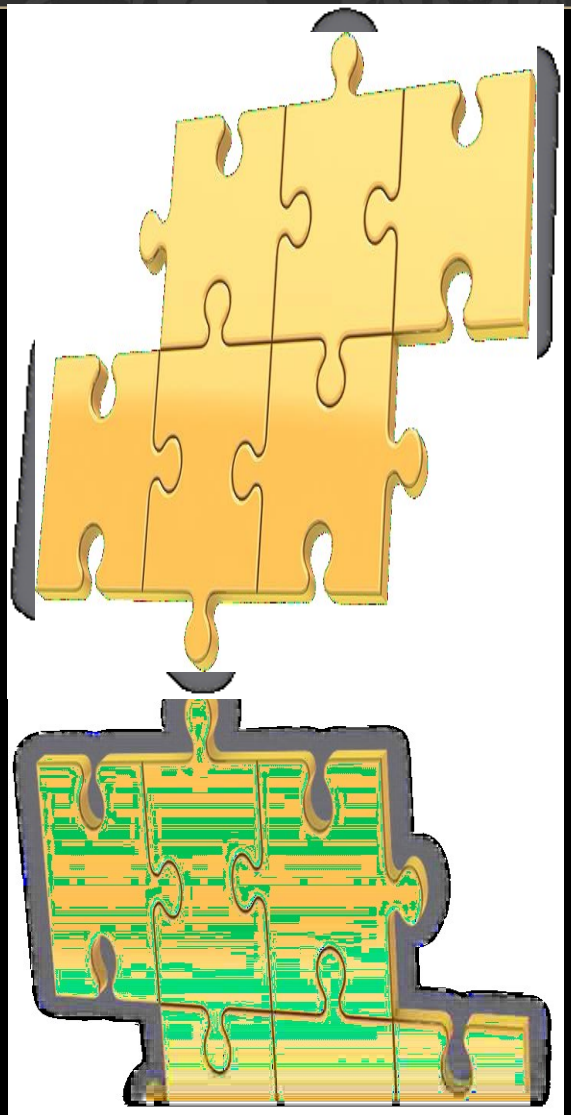
$$t_{CPU\_init} = N_{init}\cdot CPI_{init}\cdot T_{C\_init} = 1{,}6\cdot N_{init}\cdot T_{C\_init}$$

$$t_{CPU\_nou} = N_{nou}\cdot CPI_{nou}\cdot T_{C\_nou} = ((1-0{,}2\cdot 0{,}5)\cdot N_{init})\cdot 1{,}78\cdot T_{C\_init} = 0{,}9\cdot 1{,}78\cdot N_{init}\cdot T_{C\_init}$$

$$t_{CPU\_nou} = 0{,}9\cdot 1{,}78\cdot N_{init}\cdot T_{C\_init} = 1{,}6\cdot N_{init}\cdot T_{C\_init}$$

Răspuns: prin efectuarea modificărilor propuse, reducerea cu 20% a instrucţiunilor UAL nu modifica timpii de execuţie a salturilor, deci performanţa sistemului nu va creşte.

Cuprins



Informatia in Evaluarea performantelor calculatoarelor

Metrici folosite

1. Timpul de execuţie
2. Timpul CPU
3. MIPS
4. MFLOPS
5. Legea lui Amdahl

Programe reale de evaluare a performantelor

Metrici – MIPS

Timpul de execuţie este considerat cel mai important indicator de performanţă, fiind consecvent şi viabil, însă de-a lungul timpului s-au mai utilizat şi alţi indicatori, precum MIPS, MFLOPS, etc.

MIPS (Millions of instructions per second) ("Milioane de instructiuni pe secunda")

= o alternativă la timpul de execuţie,

= numărul de instrucţiuni medii pe care un calculator le poate executa pe secundă.

=> pentru un program dat, *MIPS* se defineşte după relaţia:

$$MIPS = \frac{N}{t_E \cdot 10^6} \quad (13)$$

- Scriind relaţia (13) în funcţie de frecvenţa ceasului şi numărul de cicluri pe instrucţiune, si daca tE=tCPU se obţine:

$$MIPS = \frac{t_{CPU}}{CPI \cdot t_E} \cdot \frac{f_C}{10^6} = \frac{f_C}{CPI \cdot 10^6} \quad (14)$$

Exprimând timpul de execuţie în funcţie de numărul de instrucţiuni *N* şi *MIPS*, se obţine:

$$t_E = \frac{N}{MIPS \cdot 10^6} \quad (15)$$

11. Evaluarea performantelor calculatoarelor

Metrici – MIPS (2)

În cazul în care un calculator poate executa peste un miliard de instrucţiuni pe secundă, în locul *MIPS* se poate folosi ***BIPS* (Billions of instructions per second)** sau **GIPS (G-giga)**, definit în mod similar.

Indicatorul *MIPS* este o metrică uşor de înţeles şi folosit în evaluarea performanţelor unui sistem de calcul,

însă

NU este indicată utilizarea lui *exclusivă* când se dorește
compararea performanţelor mai multor sisteme care nu aparţin aceleiaşi familii !!!
(adica nu au acelasi set de instructiuni), deoarece:

- ***MIPS* depinde de setul de instrucţiuni**
- ***MIPS* poate varia pentru programe diferite ale aceluiaşi calculator**
- ***MIPS* poate varia invers proporţional cu performanţa !**

- **MIPS nu ia in considerare timpul de executie !**

Metrici - MIPS (3)

Exemplul 08h:

Pentru sistemul de calcul din exemplul 02h -> să se calculeze *MIPS*, ştiind că ceasul sistemului are frecvenţa de 400 MHz.

Frecvenţele de utilizare a instrucţiunilor

Tip instrucţiune	*Frecvenţa*	*Cicluri ceas*
UAL	50%	1
Încărcare (load)	20%	2
Memorare (store)	10%	2
Salt	10%	3
altele	10%	2

Soluţie:

$$\text{MIPS} = \frac{f_c}{CPI \cdot 10^6} = \frac{400 \cdot 10^6}{1{,}6 \cdot 10^6} = 250$$

Metrici - MIPS (4)

Exemplul 09h:

Se dau următoarele măsurători de performanță pentru un program:

Valori obtinute pentru	*Calculator A*	*Calculator B*
Număr de instrucțiuni	8 mld	6 mld
Frecvență ceas	3,6GHz	3,6GHz
CPI	1,0	1,1

Se cere:
Care calculator are MIPS mai mare?
Care calculator este mai rapid?

Soluție:

$$MIPS_A = \frac{f_{C(A)}}{CPI_{(A)} \cdot 10^6} = \frac{3600 \cdot 10^6}{1 \cdot 10^6} = 3600$$

$$MIPS_B = \frac{f_{C(B)}}{CPI_{(B)} \cdot 10^6} = \frac{3600 \cdot 10^6}{1{,}1 \cdot 10^6} = \frac{3600}{1{,}1} = 3272{,}7272$$

$\Rightarrow$ calculatorul A are MIPS mai mare.

$$t_{E(A)} = N_A \cdot CPI_A \cdot T_{C_A} = 8 \cdot 10^9 \cdot \frac{1}{3{,}6 \cdot 10^9} \cdot 1 = 2{,}22s$$

$$t_{E(B)} = N_B \cdot CPI_B \cdot T_{C_B} = 6 \cdot 10^9 \cdot \frac{1}{3{,}6 \cdot 10^9} \cdot 1{,}1 = 1{,}66 \cdot 1{,}1 = 1{,}83s$$

$\Rightarrow$ calculatorul B este mai rapid, deși calculatorul A are MIPS mai mare.

Exemplul 0Ah: (exemplu în care MIPS nu e un indicator viabil în evaluarea performanţelor calculatoarelor)

Se pp. că un compilator care efectuează optimizarea codului este folosit pentru sistemul de calcul având frecvenţa instrucţiunilor dată în tabelul din exemplul 02h.

Frecvenţele de utilizare a instrucţiunilor

Tip instrucţiune	*Frecvenţa*	*Cicluri ceas*
UAL	50%	1
Încărcare (load)	20%	2
Memorare (store)	10%	2
Salt	10%	3
altele	10%	2

Compilatorul elimină 50% din instrucţiunile UAL, dar nu poate reduce numărul celorlalte tipuri de instrucţiuni. Dacă se cunoaşte că durata ciclului de ceas este 2,5 ns, care este valoarea MIPS şi timpul de executie al procesorului pentru codul neoptimizat şi pentru cel optimizat ?

Soluţie:

$$CPI_{neopt} = 1{,}6 \Rightarrow MIPS_{neopt} = \frac{400 \cdot 10^6}{1{,}6 \cdot 10^6} = 250$$

$$CPI_{opt} = \frac{1 \cdot (0{,}5 \cdot 0{,}5) + 2 \cdot 0{,}2 + 2 \cdot 0{,}1 + 3 \cdot 0{,}1 + 2 \cdot 0{,}1}{1 - (0{,}5 \cdot 0{,}5)} = \frac{1{,}35}{0{,}75} = 1{,}8$$

$$MIPS_{opt} = \frac{400 \cdot 10^6}{1{,}8 \cdot 10^6} = 222{,}22$$

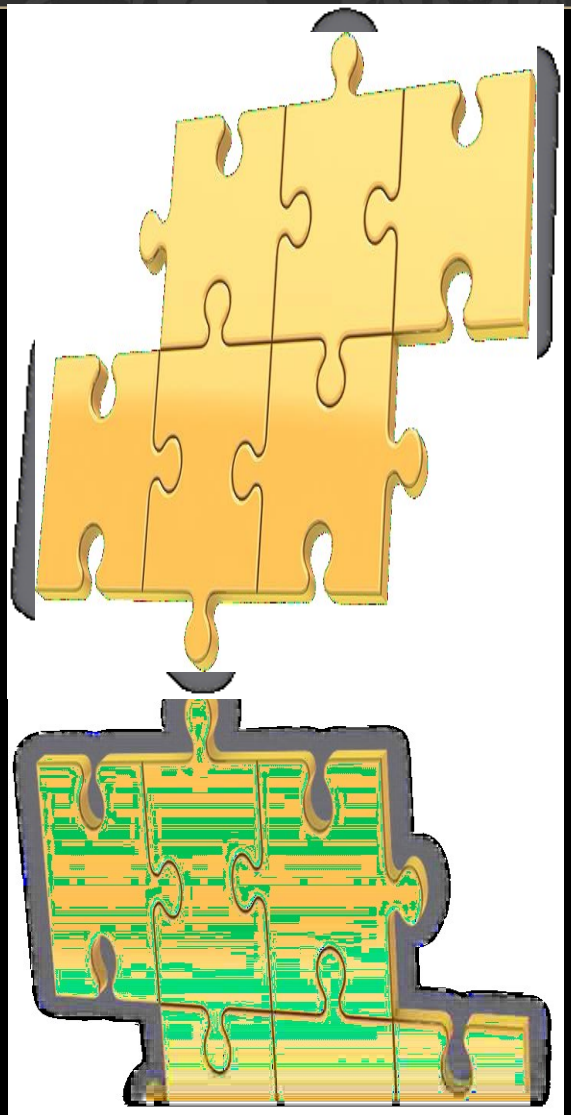
Performanţa codului neoptimizat, respectiv optimizat este:

$$t_{CPU\_neopt} = N_{neopt} \cdot CPI_{neopt} \cdot T_C = N_{neopt} \cdot 1{,}6 \cdot 2{,}5 \cdot 10^{-9} = 4 \cdot 10^{-9} \cdot N_{neopt}$$

$$t_{CPU\_opt} = N_{opt} \cdot CPI_{opt} \cdot T_C = (1 - 0{,}5 \cdot 0{,}5) \cdot N_{neopt} \cdot 1{,}8 \cdot 2{,}5 \cdot 10^{-9} = 3{,}375 \cdot 10^{-9} \cdot N_{neopt}$$

Codul optimizat este cu 15.6% mai rapid (raportat la codul neoptimizat), deşi valoarea MIPS coresp. este mai mică.

Cuprins



Informatia in Evaluarea performantelor calculatoarelor

Metrici folosite

1. Timpul de execuţie

2. Timpul CPU

3. MIPS

4. MFLOPS

5. Legea lui Amdahl

Programe reale de evaluare a performantelor

p

11. Evaluarea performantelor calculatoarelor

Metrici – MFLOPS

Metrica *MIPS* nu este adecvată procesoarelor specializate pentru ***calcule matematice*** precum procesoarele vectoriale.

- se foloseşte indicatorul ***MFLOPS*** (Millions of floating-point operations per second) sau ***GFLOPS*** (Billions of floating-point operations per second) pentru a măsura numărul operaţiilor de calcul în virgulă mobilă pe care le poate executa sistemul de calcul într-o secundă.

$$MFLOPS = \frac{N_{VM}}{t_E \cdot 10^6} \quad (15)$$

unde N_{VM} este numărul de operaţii în VM (virgulă mobilă) dintr-un program

Similar cazului metricii *MIPS*, utilizarea *MFLOPS* ridică două probleme importante:

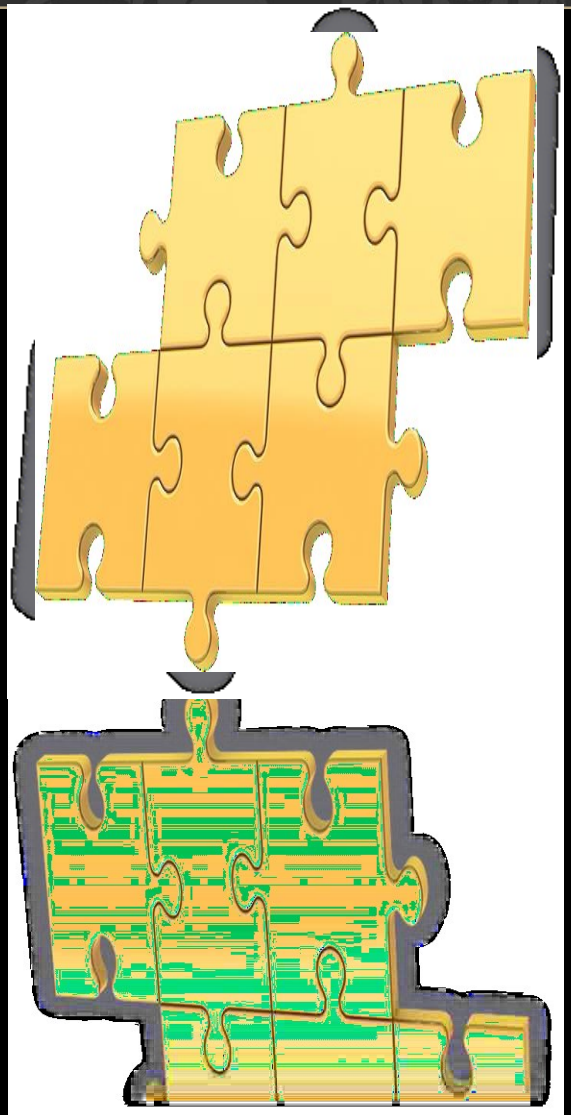
- setul de operaţii în virgulă mobilă diferă de la un calculator la altul şi
- *MFLOPS* se modifică în funcţie de diferite combinaţii între operaţii cu numere întregi şi/sau în VM

Soluţia pt rezolv acestor impedimente: fol operaţiilor normalizate în virgulă mobilă.

MIPS şi *MFLOPS* pot fi utili la compararea **sistemelor de calcul aparţinând aceleiaşi familii**, cu **acelaşi set de instrucţiuni** şi având **acelaşi ciclu de ceas**.

Studiati: GFLOPS, TFLOPS, PFLOPS

Cuprins



Informatia in Evaluarea performantelor calculatoarelor

Metrici folosite

1. Timpul de execuţie

2. Timpul CPU

3. MIPS

4. MFLOPS

5. Legea lui Amdahl

Programe reale de evaluare a performantelor

p

Legea lui Amdahl se referă la creşterea performanţei care se poate obţine prin ***îmbunătăţirea unei anumite părţi/componente*** a unui sistem de calcul.

Se presupune că se poate efectua îmbunătăţirea unui sistem de calcul prin îmbunătăţirea performanţei la execuţia unui task.

Legea lui Amdahl defineşte creşterea **performantei** ce se poate obţine prin utilizarea acestei îmbunătăţiri ca fiind:

$$\Delta v = \frac{P_{imb}}{P_{neimb}} \quad (16)$$

unde P_{imb} este performanţa obţinută utilizând îmbunătăţirea atunci când este posibil,
iar P_{neimb} este performanţa obţinută fără utilizarea îmbunătăţirii.

Imbunatatirea - >dpdv al creşterii **vitezei de executie** în funcţie de timpul de execuţie:

$$\Delta v = \frac{t_{E\_neimb}}{t_{E\_imb}} \quad (17)$$

unde t_{E_neimb} reprezintă **timpul de execuţie fără utilizarea îmbunătăţirii**,
iar t_{E_imb} este **timpul de execuţie obţinut utilizând îmbunătăţirea când e posibil**.

Creşterea vitezei depinde de:

- fracţiunea timpului de execuţie a calculatorului original în care se poate folosi îmbunătăţirea (F_{imb}), unde $F_{imb} \leq 1$,
- îmbunătăţirea obţinută prin utilizarea noului mod de execuţie,

deci creşterea vitezei care s-ar obţine dacă s-ar utiliza numai noul mod de execuţie (Δv_{imb}),
unde Δv_{imb} = raportul dintre - timpul de execuţie în modul original şi
- timpul de execuţie în modul îmbunătăţit, cu $\Delta v_{imb} > 1$.

Timpul de execuţie utilizând calculatorul original în modul îmbunătăţit va fi
timpul în care se utilizează partea nemodificată a calculatorului
+ timpul în care se utilizează partea îmbunătăţită:

$$t_{E\_nou} = t_{E\_init} \cdot \left((1 - F_{imb}) + \frac{F_{imb}}{\Delta v_{imb}} \right) \quad (18)$$

Creşterea totală a vitezei va fi raportul dintre timpii de execuţie:

$$\Delta v_{tot} = \frac{t_{E\_init}}{t_{E\_nou}} = \frac{1}{(1 - F_{imb}) + \frac{F_{imb}}{\Delta v_{imb}}} \quad (19)$$

Exemplul 0Bh:

Se presupune că un calculator ruleaza (in general) in 60% din timp operatii cu procesorul, iar in 40% din timp operatii cu hard discul si ca utilizatorul nu e multumit de performanta sistemului. Dupa studierea pietei de componente, utilizatorul a gasit 2 optiuni: prima, sa inlocuiasca procesorul cu unul de 1,5 ori mai bun pentru suma de 500 lei, iar a doua sa inlocuiasca unitatea de hard disc cu una de 2,5 ori mai buna, dar cheltuind 800 lei.

a) Care este raportul pret/performanta obtinut in ambele cazuri?

b) Care optiune va fi considerata daca se doreste o investitie cat mai mica dpdv financiar dar cu raportul pret/performanta cel mai bun?

c) Care optiune va fi consid daca se doreste obt performantei cele mai bune, indiferent de suma cheltuita?

Soluție:

a) Inlocuind procesorul, este afectat 60% din timp si se obtine:

Performanta dupa imbunatatire : $\Delta v = \frac{1}{1-0{,}6+\frac{0{,}6}{1{,}5}} = 1{,}25$ => o crestere a performantei cu 25%

Inlocuind unitatea de hard disc, este afectat 40% din timp si se obtine:

Performanta dupa imbunatatire : $\Delta v = \frac{1}{1-0{,}4+\frac{0{,}4}{2{,}5}} = 1{,}3158$ => o crestere a performantei cu 31,58%

Raportul pret/performanta: 500/125=4,00 vs 800/131,58=6,08

b) deoarece raportul pret/perform e mai bun in primul caz, se poate alege compromisul de a creste performanta doar cu 25% in loc de 31 % pentru a economisi 300 lei, deci se va alege inlocuirea procesorului

c) daca nu exista problema financiara, atunci se va alege inlocuirea hard discului deoarece performanta obtinuta este mai mare cu aproape 7% , desi raportul pret/performanta e mai ridicat cu 50%

p

Exemplul 0Ch:

Se presupune că un program rulează în 50 secunde pe un calculator, din care 40 de secunde sunt rulate operații de înmulțire.

Cu cât trebuie crescută viteza operațiilor de înmulțire pentru a rula programul de 5 ori mai rapid?

Soluție:

$t_{E\ initial}$=50s

Imbunatatirea vizeaza 40s din 50s = 80% =>F_{imb} = 0,8

$$t_{E\ \text{după îmbunătățire}} = 50s\left((1-0,8)+\frac{0,8}{n}\right) = (100s-80s)+\frac{40s}{n},$$

$n\ sau\ \Delta v_{\text{îmbunătățit}}$

de 5 ori mai rapid: $10s = \frac{40s}{n} + 10s \Rightarrow 0 = \frac{40s}{n}$

⇒ Nu există o astfel de cantitate cu care să îmbunătățim execuția operației de înmulțire pentru a obține creșterea performanței de 5 ori, dacă operația de inmulțire apare în 80% din instrucțiuni.

Exemplul 0Dh:

Se pp ca la un CPU se pot imbunatati anumite instructiuni care apar in 15% din cazuri, executandu-se de 2 ori mai rapid. Care este cresterea totala a vitezei obtinute?

Solutie:

F_{imb}=0,15; Δv_{imb}=2

$\Delta v = \frac{t_{e\_neimb}}{t_{E\_imb}} = \frac{1}{1-0,15+0,15/2} = \frac{1}{0,925} = 1,08$ => viteza a crescut cu 8%

Exemplul 0Eh:

Se presupune ca se poate imbunatati performanta unui CPU prin cresterea vitezei de executie cu un factor de 3, dar cu inconvenientul cresterii costului de 2 ori. Se mai presupune ca CPU e utilizat in doar 60% din timp, in rest avand loc alte operatii, precum cele cu dispozitivele I/O. Daca costul CPU reprezinta 1/3 din costul intregului sistem, se pune intrebarea daca e o investitie rentabila (dpdv al raportului cost/performanta) cresterea vitezei cu un factor de 3.

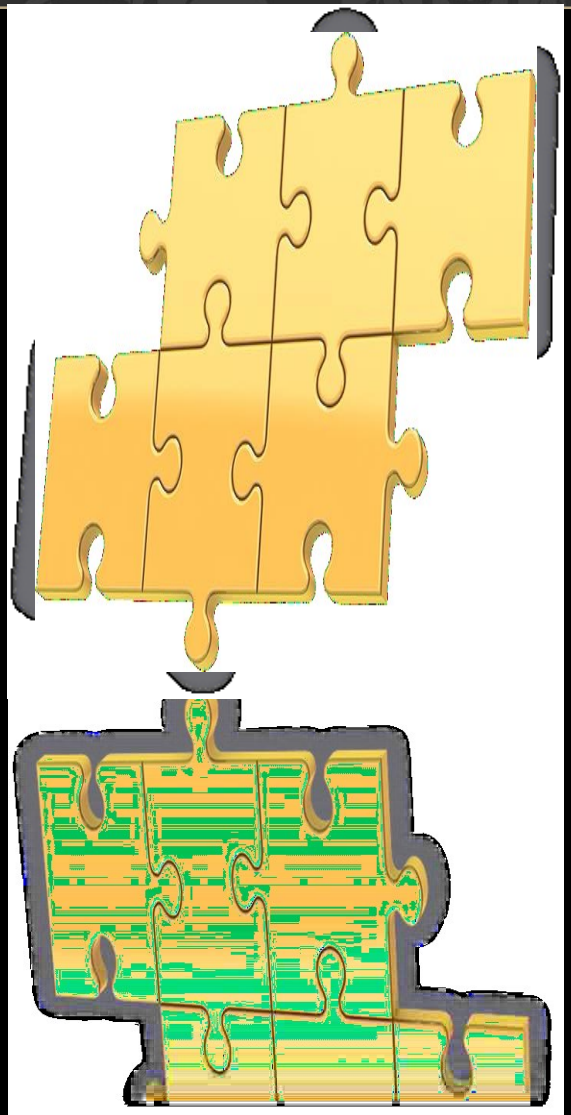
Soluţie:

Cresterea vitezei de executie va fi : $\Delta v = \frac{t_{e\_neimb}}{t_{E\_imb}} = \frac{1}{1-0,6+0,6/3} = \frac{1}{0,6} = 1,67$

=> Costul noului sistem: $\frac{2}{3} \cdot 1 + \frac{1}{3} \cdot 2 = 1,33$

deci de 1,33 ori mai mare faţă de costul calculatorului original. Astfel, creşterea costului (1,33) nu este mai mare decat creşterea performanţei (1,67) şi deci se îmbunătăţeşte raportul cost/performanţă.

Cuprins



Informatia in Evaluarea performantelor calculatoarelor

Metrici folosite

1. Timpul de execuţie
2. Timpul CPU
3. MIPS
4. MFLOPS
5. Legea lui Amdahl

Programe reale de evaluare a performantelor

p

11. Evaluarea performantelor calculatoarelor

Programe reale de evaluare a performantelor

În general, performanţa unui sistem de calcul nu se poate caracteriza printr-o singură metrică, deoarece aceasta depinde de interacţiunile diferitelor componente ale sale.

Pe măsură ce arhitectura calculatoarelor evoluează, devine tot mai dificil de comparat performanţa diferitelor sisteme doar prin specificaţiile date de producător => compararea performanţelor prin
rularea unor ***programe de testare a performanţelor (benchmarks) standardizate***

=>s-au dezvoltat diferite programe specializate (numite **benchmark-uri**) pt evaluarea performanţelor unor componente sau ale unor sisteme de calcul:

- UCP (Unitatea centrală de procesare)
- unitatea aritmetică în virgulă fixă / mobilă
- sistemul de memorie
- sistemul de I/O
- sistemul de operare

Metricile tradiţionale precum MIPS sau MFLOPS uneori pot fi eronate şi nu reflectă factori relevanţi precum timpul de execuţie al aplicaţiilor, utilizarea memoriei, energia consumată.

Aplicaţiile de tip benchmark suferă şi ele unele limitări care împiedică compararea corectă a mai multor sisteme. Există totusi o metodologie pe bază de nuclee (numite kernele) şi diferite aplicaţii ce furnizează estimări foarte pertinente asupra performanţei procesorului, deoarece sunt **standardizate**
=> se folosesc aplicaţii complete sau chiar seturi de aplicaţii
ce permit inclusiv măsurarea timpului de utilizare a memoriei, energia consumată de procesor, ...

11. Evaluarea performantelor calculatoarelor

Programe reale de evaluare a performantelor (2)

Categorii pt programele de test [Patterson09]:

1. ***Programe reale***: ex :compilatoare de C, programe de procesare text, programe de proiectare asistată (Spice).
Testarea: în condiţii similare privind intrările, ieşirile şi setarea opţiunilor.
Dezavantaj: uneori se pot întâlni probleme de portabilitate, mai ales datorită dependenţei de S.O. folosit
2. ***Aplicaţii modificate*:** se fol aplicaţiile reale ca blocuri componente ale unor programe de testare. Modificările constau în eliminarea unor componente (de ex eliminarea componentelor de I/O dacă programul de test este destinat performanţelor UCP) sau prin introducerea unor componente care măresc gradul de portabilitate al programelor.
Se fol Script-uri pt a simula programe de aplicaţie şi cu scopul de a reproduce o *comportare interactivă* (de ex prin afişări pe ecranul calculatorului) sau pt a simula *interacţiunea multi-utilizator* produsă pe un sistem de tip server.
3. ***Nuclee (kernels)*** din programe reale: s-au extras porţiuni semnificative (=nuclee) din programe reale, pt a fi folosite ca rutine de test. Exemple: Livermore Loops, Linpack.
4. ***Toy benchmarks*** (programe de test amuzante – jucărie- nu sunt ft concludente): sunt programe scurte (maxim 100 de linii de cod) care produc un *rezultat cunoscut înainte de rulare,* folosite doar pentru a testa *viteza de execuţie*.
Exemplu: Sieve of Erastosthenes, Puzzle, Quicksort.
5. ***Programe de test sintetice (synthetic benchmarks)***: programe create artificial, dar asemănătoare ca scop cu programele de test de tip "kernels"-> testarea: prin operaţii şi operanzi specifici, cu frecvenţa medie de apariţie în programele reale. Exemple: *Drystone* şi *Whetstone*. Programul de test Dhrystone -> analiză statistică , fiind fol pt testarea performanţelor la aritmetica cu numere întregi şi pt modul de transmitere a parametrilor la apelul unor funcţii; testul Whetstone = o colecţie de coduri ce testează performanţa la rularea unor biblioteci matematice în virgulă mobilă; Programele ca Drystone şi Whetstone - *testează amănunţit doar UCP*, dar nu testează performanţa celorlalte componente ale sistemului (precum unităţile de disc, dispozitivul de afişare, etc).

11. Evaluarea performantelor calculatoarelor

Programe reale de evaluare a performantelor (3)

Standarde in industrie

Business Applications Performance Corporation (BAPCo)
-> SYSmark 2012 (performanta generala a intregului sistem), TabletMark 2013 (pt dispozitive tactile), MobileMark 2012 (bateria si performanta sistemului)
- de studiat : http://www.ginfo.ro/revista/11_3/tehno.pdf

Embedded Microprocessor Benchmark Consortium (EEMBC)("embassy")
- in domeniul sistemelor cu calculator integrat (Embedded Systems)
-> exista 5 tipuri de seturi de programe de test în funcţie de domeniile de aplicaţie:
auto şi industrial, bunuri de consum, interconectare, automatizări de birou, telecomunicaţii

Standard Performance Evaluation Corporation (SPEC), cu SPECint si SPECfp
- in 1988 - corporaţia SPEC (non-profit) – scop: *stabilirea, menţinerea şi garantarea* de seturi standardizate pt programe de test
– initial cuprindea firmele HP, DEC, MIPS şi Sun; a crescut în timp, incluzând în prezent peste 60 de companii
- SPEC nu rulează programe de test (nu realizează testare de maşini de calcul), ci dezvoltă programe de test, analizează şi publică rezultatele obţinute şi transmise către SPEC de membrii organizaţiei şi alte instituţii licenţiate

Transaction Processing Performance Council (TPC)
- măsoară capacităţile unui sistem în realizarea tranzacţiilor (accesari şi actualizări de baze de date)
- Primul program de test: 1985 TPC-A;
intre timp: multe alte variante, de ex. TPC-W pt testarea performanţelor tranzacţiilor pe bază de Web.

Coremark – pentru sisteme embedded

11. Evaluarea performantelor calculatoarelor

Programe reale de evaluare a performantelor (4)

- *Linpack* = o colectie de rutine bazate pe algebra liniara si metode Gaussiene
  - de 2 ori pe an se publica pe *www.top500.org* o lista cu primele 500 computere care au cea mai ridicata performanta pe baza acestui benchmark [Patterson 2013]
  -> primul sistem din lista top500 e considerat "**cel mai rapid sistem din lume**"
- *SPECrate* = o metrica bazata pe debit – bazata pe benchmarkuri din suita SPEC CPU (2006 de exemplu)
  - masoara paralelismul la nivel de task (desi ruleaza mai multe copii ale unui program simultan, nu exista comunicare intre taskuri)
- *SPLASH* si *SPLASH 2* (Stanford Parallel Applications for Shared Memory) – Universitatea Stanford
  - similar SPEC CPU, dar fol. 2 seturi de date si aplica benchmarkuri paralele (kernele si aplicatii)

SPEC (*System Performance Evaluation Cooperative*) – o organizatie sprijinita de industria producatorilor de sisteme pentru a asigura ***seturi standardizate de benchmarkuri***

In 1989: SPEC a creat un benchmark pt evaluarea performantei CPU – "SPEC89" - a evoluat in 5 generatii
...
In 2006: SPEC a creat "SPEC CPU2006" – un set de ***12 benchmarkuri*** pentru operatii intregi (CINT2006) si ***17 benchmarkuri*** pentru operatii in virgula mobila (CFP2006). [Patterson2013]

11. Evaluarea performantelor calculatoarelor

Programe reale de evaluare a performantelor (5)

SPEC2006 benchmark description	SPEC2006	SPEC2000	SPEC95	SPEC92	SPEC89
GNU C compiler					gcc
Interpreted string processing			perl		espresso
Combinatorial optimization		mcf			li
Block-sorting compression		bzip2		compress	eqntott
Go game (AI)	go	vortex	go	sc	
Video compression	h264avc	gzip	ijpeg		
Games/path finding	astar	eon	m88ksim		
Search gene sequence	hmmer	twolf			
Quantum computer simulation	libquantum	vortex			
Discrete event simulation library	omnetpp	vpr			
Chess game (AI)	sjeng	crafty			
XML parsing	xalancbmk	parser			
CFD/blast waves	bwaves				fpppp
Numerical relativity	cactusADM				tomcatv
Finite element code	calculix				doduc
Differential equation solver framework	dealII				nasa7
Quantum chemistry	gamess				spice
EM solver (freq/time domain)	GemsFDTD			swim	matrix300
Scalable molecular dynamics (~NAMD)	gromacs		apsi	hydro2d	
Lattice Boltzman method (fluid/air flow)	lbm		mgrid	su2cor	
Large eddie simulation/turbulent CFD	LESlie3d	wupwise	applu	wave5	
Lattice quantum chromodynamics	milc	apply	turb3d		
Molecular dynamics	namd	galgel			
Image ray tracing	povray	mesa			
Spare linear algebra	soplex	art			
Speech recognition	sphinx3	equake			
Quantum chemistry/object oriented	tonto	facerec			
Weather research and forecasting	wrf	ammp			
Magneto hydrodynamics (astrophysics)	zeusmp	lucas			
		fma3d			
		sixtrack			

[Henessy2009]

- programele de test SPEC = *seturi* de aplicaţii pentru testarea procesoarelor Avantaj seturi: eventuala slăbiciune a uneia dintre aplicaţii poate fi compensată rulând celelalte aplicaţii din setul etalon

- există 5 generaţii de seturi de programe de test, pt operare în virgulă fixă (INT - întregi) şi virgulă mobilă (reale, FP - Floating Point): SPEC89, SPEC92, SPEC95, SPEC2000, SPEC2006 (vers curenta **V1.2** - Sept 2011)

- SPEC CPU2006 conţine 2 serii de programe de test: CINT2006 , CFP2006 pt măsurarea şi compararea performanţelor la calcule cu întregi (12 seturi), respectiv în virgulă mobilă (17 seturi).
Cele 2 -> măsoară performanţa pt procesor, pt arhitectura memoriei şi pt compilator.

11. Evaluarea performantelor calculatoarelor

Programe reale de evaluare a performantelor (6)

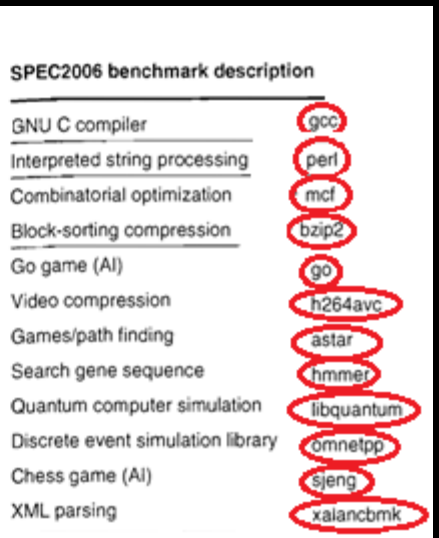
SPEC a decis sa raporteze performanta SC la ***un singur numar*** – pt a sintetiza toate cele 12 benchmarkuri de tip intreg
=> s-a divizat timpul de executie al unui CPU de referinta la timpul de executie obtinut de sistemul testat = "normalizare"
=> **valoarea *SPECratio - direct proportionala cu performanta sistemului (avantaj major)***
Valoarea rezultata din masuratorile pe CINT2006 sau CFP2006 se obtine din media geometrica a valorilor SPECratio

Exemplu:
Benchmarkurile SPECINTC2006 rulate pe un procesor Intel Core i7 920 la 2.66 GHz [Patterson2013]
- **timpul de executie (rel. 8)** = produsul a 3 factori:
  Nr de instructiuni (masurat in miliarde) = I coloana
  Nr de cicluri pe instructiune adica CPI (se observa ca CPI variaza cu un factor >5) = a II-a coloana,
  Durata ciclului de ceas masurata in nanosecunde (e aceeasi pt toate liniile din tabel Tc= 0.376 (sec x 10^{-9})).

Ultima coloana: **SPECratio** = timpul de referinta (furnizat de SPEC – coloana IV) si divizat la timpul calculat cu relatia (8)
=> **SPECINTC2006** va fi media geometrica a valorilor de pe ultima coloana (SPECratio) = **25.7** [Patterson2013]

Concluzie:
La compararea a 2 SC prin SPECratio, se va folosi **media geometrica**
=> va furniza acelasi raspuns relativ indiferent de calculatorul utilizat ac referinta (adica pentru normalizare)
- daca s-ar folosi media aritmetica rezultatele ar varia functie de SC de referinta

Nr instruct. x10^9	CPI	Timp executie (sec)	Timp de referinta (sec)	SPECratio
794	1.20	358	8050	22.5
2252	0.60	508	9770	19.2
221	2.66	221	9120	41.2
2390	0.70	629	9650	15.4
1274	1.10	527	10490	19.9
3793	0.50	713	22130	31.0
1250	1.00	470	7020	14.9
2616	0.60	590	9330	15.8
659	0.44	109	20720	190.0
367	2.10	290	6250	21.5
1948	0.80	586	12100	20.7
1045	0.70	275	6900	25.1
Media geometrica				25.7



"Fallacies and Pitfalls" [Patterson2013]

Pitfalls = "easily made mistakes"

= "generalizations of principles that are true only in a limited context"

"Pitfall: Expecting the improvement of one aspect of a computer to increase overall performance by an amount proportional to the size of that improvement."

-> *Legea lui* **Amdahl**

"Pitfall: Using a subset of the performance equation as a performance metric."

"commonly held misconceptions = *fallacies" (aberatii)*

"Fallacy: Computers at low utilization use little power" (consum proportional)

"Fallacy: Designing for performance and designing for energy efficiency are unrelated goals (independenta

11. Evaluarea performantelor calculatoarelor

Programe reale de evaluare a performantelor (7)

Open source benchmarks [Wikipedia]

DEISA Benchmark Suite: pt aplicatii stiintifice
Dhrystone, Fhourstones: aritmetica intreaga
Whetstone: aritmetica in virgula mobila
HINT: performanta procesorului si a memoriei
Iometer: subsistemul I/O
Linpack: masoara FLOPS
NBench: performanta pt numere intregi, in virgula mobila si memorie (sintetic)
Geek Benchmark: performanta pt numere intregi, in virgula mobila si memorie
POV-Ray: redare 3D
TATP Benchmark: pt tranzactii in telecomunicatii
TPoX: tranzactii pe baze date
Rodinia: arhitecturi paralele (acceleratoare)
Parsec: sisteme cu arhitecturi de memorie paralele
Splash2: arhitecturi paralele
STREAM: banda memoriei
LLCbench: (Low Level Architectural Characterization Benchmark Suite) perform procesorului si a memoriei

Altele:

BAPCo: MobileMark, SYSmark, WebMark
Windows System Assessment Tool
Futuremark: 3DMark, PCMark
iCOMP – fol de Intel
Performance Rating – fol de AMD si Cyrix
Sunspider – viteza browserului

Worldbench
PiFast
SuperPrime
Super PI, Hyper pi
wPrime, IntelBurnTest, Prime95, Montecarlo superPI, OCCT , y-cruncher

Daca un SC e capabil sa calculeze pi (pana la pozitia 32 milioane dupa punctul zecimal) fara nici o greseala, se considera a fi stabil dpdv al memoriei si procesorului; testul poate dura ore in loc de minute pe unele sisteme

- super pi este calcul pt un singur thread, deci nu e relevant la evaluarea performantelor sistemelor multiprocesor

=> hyper pi – pt threaduri multiple

p

Programe reale de evaluare a performantelor (8)

Benchmarks = set de programe reprezentative pt evaluarea sistemelor de calcul

- aceste programe efectueaza anumite operatii si masoara timpul în care acestea au fost efectuate:

Rezultatele sunt comparate cu cele obþinute de un sistem de referinta,

iar sistemul testat primeste un punctaj in functie de rezultat (=scor)

- pt a det punctajul corespunzator unei aplicatii, se compara timpul de care a avut nevoie sistemul pt a îndeplini sarcina cu timpul în care a îndeplinit sarcina un sistem considerat *sistem etalon*

- Punctajul corespunzator unei categorii este determinat **calculând media geometrica** a punctajelor obtinute pentru fiecare aplicatie din categoria respectiva.

=> Punctajul general – det. tot pe baza unei medii geometrice, dar a punctajelor coresp. fiecarei categorii

O masura a performantei:

timpul de executie al unui set reprezentativ de programe

- timpul de executie poate fi: timpul total de executie

media aritmetica sau geometrica a timpilor de executie

Sintetizarea si compararea performantei unui grup (set) de benchmarkuri

- daca nu exista o relatie unitara intre masuratorile sau rezultatele obtinute, nu se poate defini performanta relativa intre 2 sau mai multe SC pe baza timpului de executie (pt fiecare program in parte):

- exemplu: un program ruleaza pe calculatorul A mai rapid decat pe B, dar un alt program ruleaza pe calculatorul B mai rapid decat pe A => setul resp nu este relevant

p

Programe reale de evaluare a performantelor (9)

- timpul TOTAL de executie - cea mai simpla metoda
  - se spune ca un calculator A este de n ori , unde n=tE(B)/tE(A) , mai rapid decat calculatorul B pt un set de programe
- media aritmetica a timpilor de executie: $MA = \frac{1}{n} \sum_{i=1}^{n} t_{Ei}$, unde t_{Ei} este timpul de executie al programului i din setul total cu n programe de test

- media aritmetica ponderata a timpilor de executie
  - se fol daca exista anumite programe care se executa mai des din set
  - fiecare program are atribuita o pondere ce indica frecventa sa de executie
  - alegerea ponderilor se face a.i. timpii de executie ponderati ai fiecarui program sa fie egali (pe un calculator de referinta)

	tE(A)	tE(B)	tE(A normalizat la A)	tE(B normalizat la A)	tE(A normalizat la B)	tE(B normaliza la B)
Progr 1	1	10	1	10	0,1	1
Progr 2	1000	100	1	0,1	10	1
Media aritm	500,5	55	1	5,05	5,05	1
Media geom	31,6	31,6	1	1	1	1

Se pare ca A este mai rapid ca B

Se pare ca B este mai rapid ca A

11. Evaluarea performantelor calculatoarelor

Programe reale de evaluare a performantelor (10)

La normalizare: se consid media timpilor de executie normalizati
- daca se fol media aritmetica a timpilor de executie, rezultatul va depinde de alegerea calculatorului de referinta !

- media geometrica are avantajul ca este independenta de seria datelor folosite la normalizare: $MG = \sqrt[n]{\prod_{i=1}^{n} t_{Ei}}$

=> rezultatul va fi acelasi, corect, indiferent de calculatorul ales la normalizare
- dezavantaj: nu anticipeaza performantele

Programe de evaluare a performantelor:

- se prefera folosirea unui set de aplicatii reale ca programe de evaluare
- programe artificiale (sintetice) de evaluare (kernele):
  se creaza un program in care frecventele de executie ale instructiunilor sunt aceleasi cu cele dintr-un set de programe de evaluare

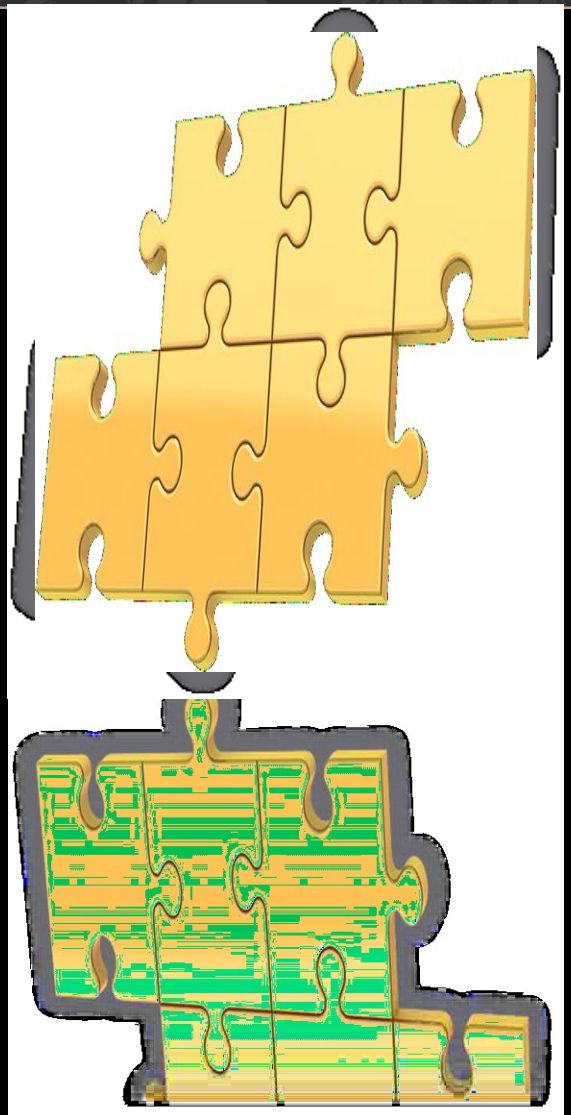
11. Evaluarea performantelor calculatoarelor

Programe reale de evaluare a performantelor (11)

Diferiţi **utilizatori** pot fi interesaţi de diferite alte aspecte, dintre care :
- consumul de energie – să fie unul cât mai redus;
- dimensiune redusă şi greutate mică – mai ales la sistemele portabile;
- costul sistemului – uneori se raportează viteza sistemului la cost;
- performanţa per watt – în special la sistemele paralele.

Alţi factori de calitate care se urmăresc la evaluarea performanţelor unui SC sunt:
- generalitatea – gama de aplicaţii ce se pot rula pe o arhitectură;
- simplitatea în utilizare – uşurinţa dezvoltării de programe pentru arhitecturile respective;
- expandabilitatea – uşurinţa cu care se poate extinde o arhitectură (cu procesoare, memorii, dispozitive de I/E);
- compatibilitatea – în ce măsură pot fi executate (pe noile arhitecturi) programele dezvoltate pentru arhitecturile precedente din aceeaşi familie;
- fiabilitatea – probabilitatea de apariţie a defectelor sau timpul mediu între defecte.

Pentru **producători,** este mai importantă energia consumată (ce determină durata de viaţă a bateriei) decât puterea consumată;
se ţine cont de managementul puterii.



Concluzii

Concluzii

- Evaluarea performantelor calculatoarelor nu trebuie sa aiba legatura cu informatia vaga, incompleta sau apelul la popularitate.

- Metricile uzuale folosite la evaluarea performantelor PC-urilor sunt:
Timpul de execuţie , Timpul CPU, MIPS, MFLOPS, Legea lui Amdahl

- La folosirea unor programe reale de evaluare a performantelor PC-urilor trebuie considerate urmatoarele aspecte:

- se prefera folosirea unui set de aplicatii reale ca programe de evaluare

- programe artificiale (sintetice) de evaluare (kernele):
se creaza un program in care frecventele de executie ale instructiunilor sunt aceleasi cu cele dintr-un set de programe de evaluare

Bibliografie principala

[Barr2005] – Mostafa Abd-El-Barr, Hesham El-Rewini

– "Fundamentals of Computer Organization and Architecture", 2005

[Baruch2000] - Zoltan Baruch

– "Arhitectura calculatoarelor", Editura Todesco, 2000

[Brey1997] - Barry B. Brey

- "The Intel Microprocessors", 4th edition, 1997

[Henessy2009] - John Hennesy, David Patterson

– "Computer Architecture – A quantitative Approach" , 2009, 5th edition

[Hide2001] - Randall Hide

– "The Art of Assembly Language", beta edition

[Lupu2012] – Eugen Lupu, Simina Emerich , Anca Apatean

– "Initiere in Limbaj de Asamblare x86. Lucrari practice, teste si probleme", Ed. Galaxia Gutenberg, 2012

[Mueller2012] - Scott Mueller

– "Upgrading and Repairing PCs", 20th edition, 2012

[Null2003] - Linda Null, Julia Lobur

– "The essentials of Computer Organization and Architecture", 2003

[Patterson2009] – David Patterson, John Hennesy

– "Computer Organization and Design – the hardware/software interface", 4th edition, 2009

[Patterson2013] – David Patterson, John Hennesy

– "Computer Organization and Design – the hardware/software interface", 5th edition, 2013

[Tarnoff2007] - David Tarnoff

– "Computer Organization and Design Fundamentals", editia intai revizuita, 2007